Simply Statics

A Project Undertaken By
Group 6: Albert Li, Dominic Croce, and Jake Ramirez

Abstract

In introductory structural analysis, students are often taught basic methods for solving truss and beam systems, such as the method of joints for trusses or the integration method for beams. While these are fairly simple to understand and iteratively implement, the chance to commit mistakes drastically increases as either the number of joints in a truss system increase or the loading conditions on a beam become too complicated. When mistakes are identified, oftentimes they have less to do with the student's intuition about beam or truss structures and more to do with simple math or algebra mistakes. This wastes time and clouds the student's conceptual understanding of the material. To remedy this situation, we have designed a combined visual beam and truss solver to provide basic information on various truss structures and beam loading conditions within the bounds of an introductory static mechanics course. The aim of this project is to provide deeper insight with *visual*, rather than *mathematical*, tools into the behavior of these systems to aid in students' understanding of static structures by bypassing complicated derivations that are often dense compared to the system being analyzed.

General Description

Simply Statics is a graphical structural mechanics program that aims to aid students' ability to intuitively visualize the effects of arbitrary loading conditions on various support conditions. It allows the user to very quickly modify and reanalyze structural systems in a way that would be very cumbersome to do by hand. We believe this tool to be very useful as an educational supplement to most introductory static mechanics courses, or for any student curious about basic structural analysis.

There are two main parts to this application: the beam solver, and the truss solver. Both of them are available on a single GUI, which allows the user to switch between them at ease.

Beam solver functionalities:

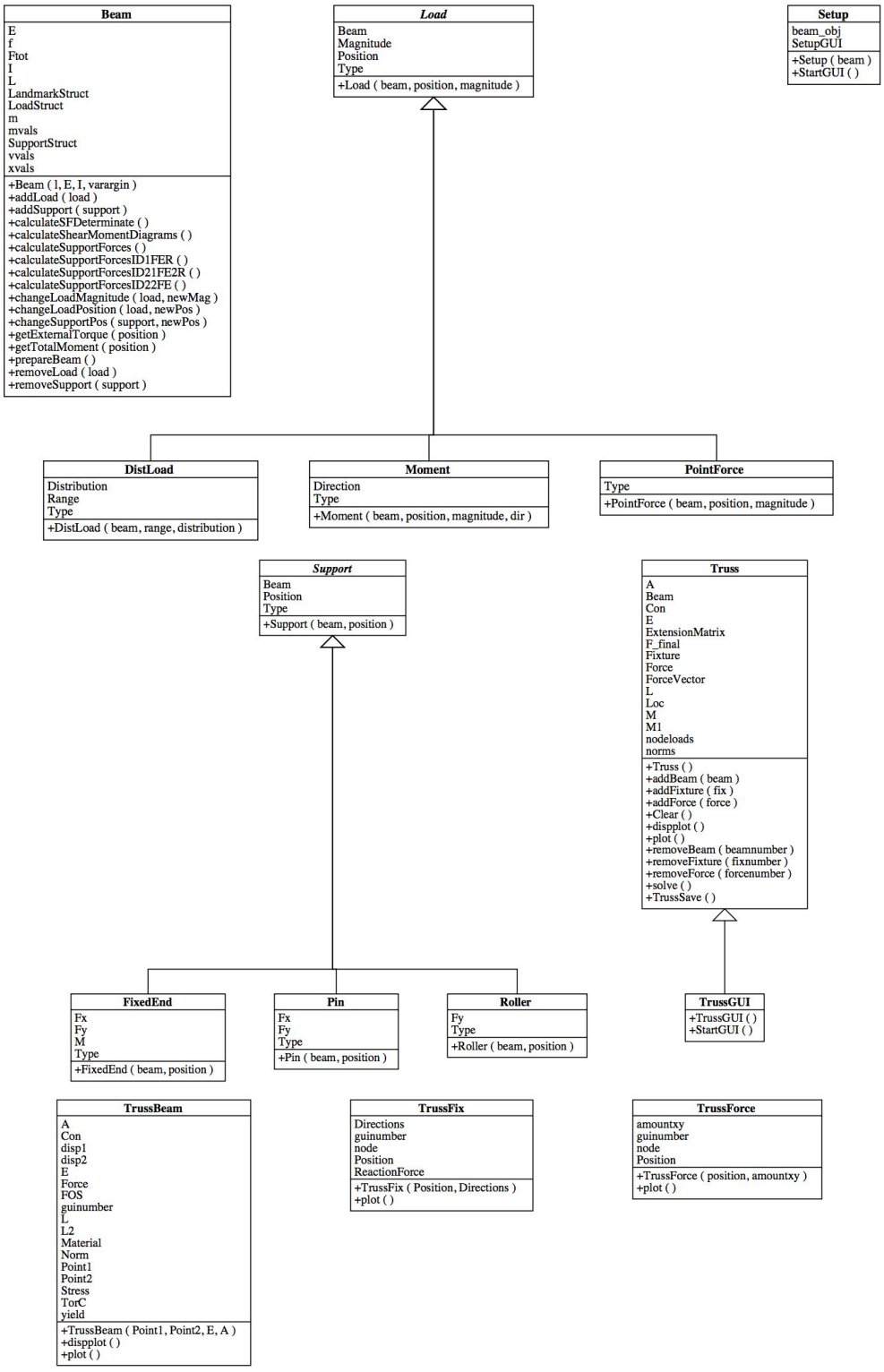
- Solves almost all cases for beams indeterminate to the second degree or lower with transverse loading conditions
- Can have any arbitrary number of loading conditions, and allows for loads going up/down, cw/ccw, and arbitrary function inputs for distributed loads
- Supports different beam cross-sections and materials
- Generates the shear/moment diagrams and indicates max moment/shear
- Supports the ability to selectively clear either loads or supports

Truss solver functionalities:

- Solves any truss that is properly constrained to give accurate results
- Considers displacements so forces are much more accurate than solving a truss using method of joints/ method of sections
- Accepts beams of various materials and cross sections in the same truss
- Quickly create trusses using easy beam mode and checking any two boxes
- Right click on any check box to instantly add forces and supports
- Generates a plot of the solved truss
- An all purpose feed for:
    - displaying information by left clicking on any graphical object
    - Notifying the user of what mode the solver is in
    - Gives overall solution information for quickly accessing the solution
- Easily clear the whole truss or right click on specific features to delete them for easy truss editing.

Project Structure

UML Diagram



The Beam class is the main class of the beam solver. It represents the beam object and inherits from handle. It has many properties, including Young's Modulus and the Moment of Inertia of its cross section and also data about its loading and support conditions stored in structs. It has the ability to add, remove, and change loads and supports by looping through the structs. The user can also determine the degree of indeterminacy of the beam and calculate torque about a point with and without support reactions. Finally, it has 5 methods for solving certain types of beam conditions by setting up a solutions matrix and for the indeterminate cases by analytical integration of distributed loads using Matlab's symbols toolbox, so there is no numerical drift at all, even for extremely complicated loadings. There is one more method that calculates the shear and moment diagrams by looping through all support reactions and loads and numerically stores the plot data as properties of the beam (xvals, vvals, mvals).

There are 8 other beam solver classes, 6 of which inherit from the Load and Support classes. Every support and load is represented by a distinct object with properties that relate to that type. For example, a Roller support does not have a Fx property because it is not constrained in that direction, while a FixedEnd does. There are also protections against initializing these objects in the wrong way, such as prevention from initializing a FixedEnd in the middle of a beam.

The Truss Solver is similarly structured like the Beam Solver where the Truss class is the main class. Its main properties, Beam, Fixture, and Force, are structs that store all of the TrussBeam's, TrussFix's, and TrussForce's added to it. TrussBeam, TrussFix, and TrussForce store properties of the beams, supports, and forces respectively, contain methods for basic calculations of some dependent properties, and contain the methods for plotting each of the object individually. For example a TrussBeam would have properties for its two end points, its material, and its cross sectional area and dependent properties like stress to be calculated after

the truss is solved. The Truss class's methods include methods to add and remove beams, supports, and forces, methods that calculate intermediate steps of solving the truss matrix, methods that support GUI functionality, and a main method that solves for the displacements of all the nodes using the previously calculated dependent properties. Every truss based class inherits from handle because everything needed to be referenced by handle and we didn't want to create separate by value instances of objects.

The GUI structure consists of two tabs, one for the beam solver and one for the truss solver. The beam solver begins with the Input Data tab, where the user selects material, cross section, and inputs a length. From there, a plot is generated to represent the beam, and the user may add blue supports and red loadings. This plotting proved to be the most involved part of the beam solver, as it required scaling loadings relative to each other and determining proper direction for plotting arrows. The user may then calculate the moment and shear diagrams in the final panel. The design flows left to right, starting with the creation of the beam, next to the loadings and supports, and finally with the calculation.

The Truss Solver tab features an input panels for creating beams, supports and forces, a feed to display information, and display to visualize the truss solution and displacement. A user would be constantly interfacing with the interactive display to add all of the features of the truss and then solving the truss to output solution information in the feed. The feed was a useful feature because tooltips cannot be created for graphical objects like lines. Because of this particularity we added both context menus and button down functions for all of the graphical objects as they were created. The button down functions (left click) serve to send information about the graphical object to the feed and the context menus (right click) server to offer a delete button for graphical objects for easy truss editing.

Usage and Operation

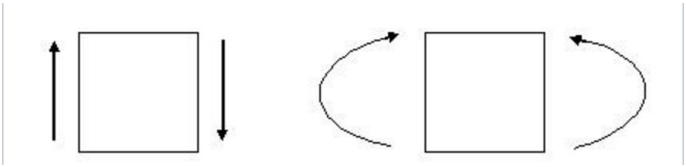
To start the GUI, type the command: “Setup.StartGUI” in the command line without the quotes.

Beam Solver:

To use the beam solver, first be sure that the “Beam Solver” tab is selected. Then, use the dropdown menus to choose a material and cross section, input a length, and click on “Create Beam.” From there, add supports and loadings in the order desired. For the Distributed Load option, input a function of x in Matlab syntax. For example, “x squared” would be “x.^2” and “three x” would be “3*x.” It is critical that exponent operations are performed element-wise! “x^2” is incorrect notation and will throw an error. The variable must be a lowercase x. Clicking calculate shows moment/shear diagrams for valid inputs and updates the info feed.

Once calculated, the user has option to clear everything, loadings, supports, diagrams, or to calculate again. The user may add more loadings and/or supports, click calculate again, and compare the moment and shear diagrams from the previous setup and the current one.

A limitation of the beam solver is the inability to delete singular supports and loadings. The user must clear all supports or loadings at once. While an inconvenience, it is easy to add desired supports or loadings again. Additionally, if a user misplaces a value, there is currently no capability to change those values on demand. Instead, the user must clear loads and generate the setup yet again. Our convention for positive shear and moment is in the picture below.



Positive Shear and Moment Convention

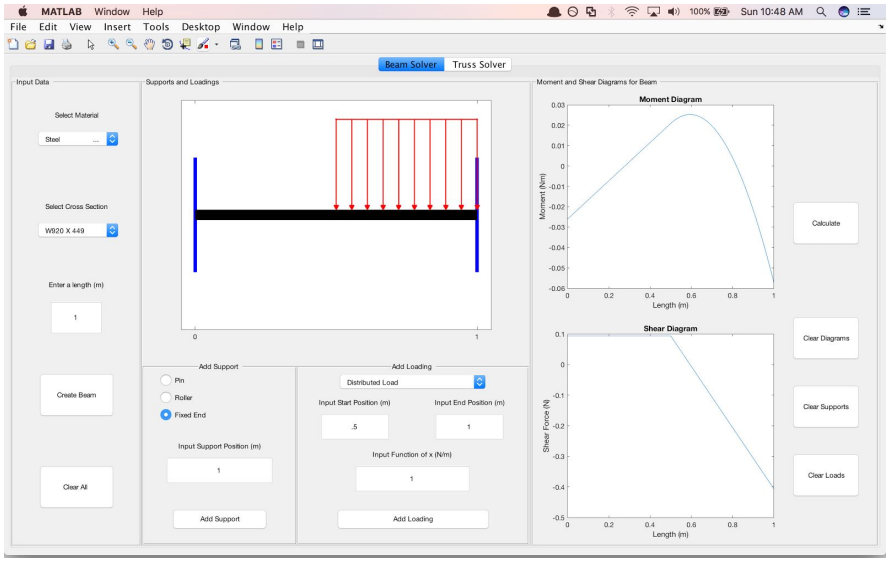
Truss Solver:

To use the truss solver, first be sure that the “Truss Solver” tab is highlighted. Then input the proper material and cross sectional area for the beams to use in the truss. Turn on “Easy Beam Mode” to easily place beams by checking two distinct checkboxes. Add supports by selecting the support directions in the support panel and then right click on the check boxes and choose the “Add Support” option from the dropdown menu. The same goes for Forces: set parameters and then right click and choose “Add Force”. The user can also numerically add supports and forces by inputting a position “x,y” into each respective position box and then pressing the “Add Support” or “Add Force” buttons. If one beam, support, or force is placed incorrectly, right click on the graphical feature and select “delete ____”. If the whole truss needs attention press the button “clear all features”. To ensure a placed beam has the proper material, a support is a pin or roller, or a force is the correct magnitude, left click on the graphical feature to display its information in the feed. Once the Truss is ready, hit the solve button to display a graphical representation of the solution and to get information in the feed. Like before, the user can left click on generated beams to get information about the solved beam like the stress and factor of safety. Supports will now display reaction forces when left clicked. To edit the truss again, clear the solution plot and make modifications as needed.

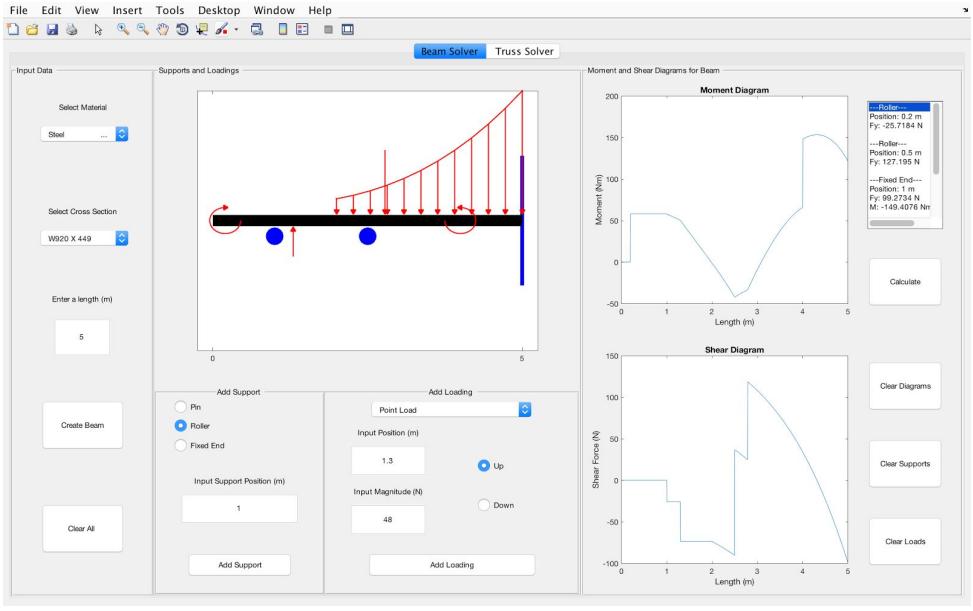
One limitation in the truss solver is that the user cannot add nodes midbeam. The only interactions take place at nodes that the beam knows it’s a part of (start and end point). So with a horizontal beam from (0,0) to (2,0) and then try to form a connection with another beam at (1,0), the two beams won’t recognize that the user tried to generate a new node and the second beam would in essence be floating in space. The same effect occurs for beams that cross the same path. They will not interact with each other and operate as if they are completely distinct.

Examples

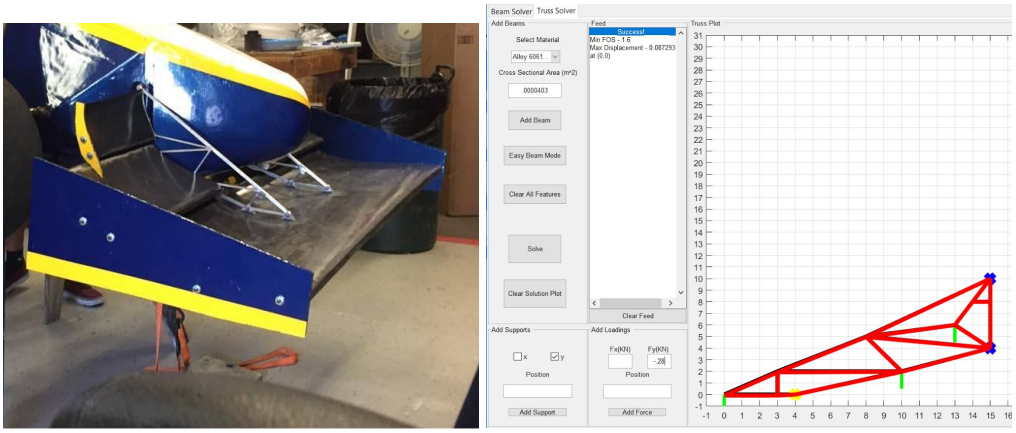
Beam Solver: Actual Final Exam Problem (Fall 2016)



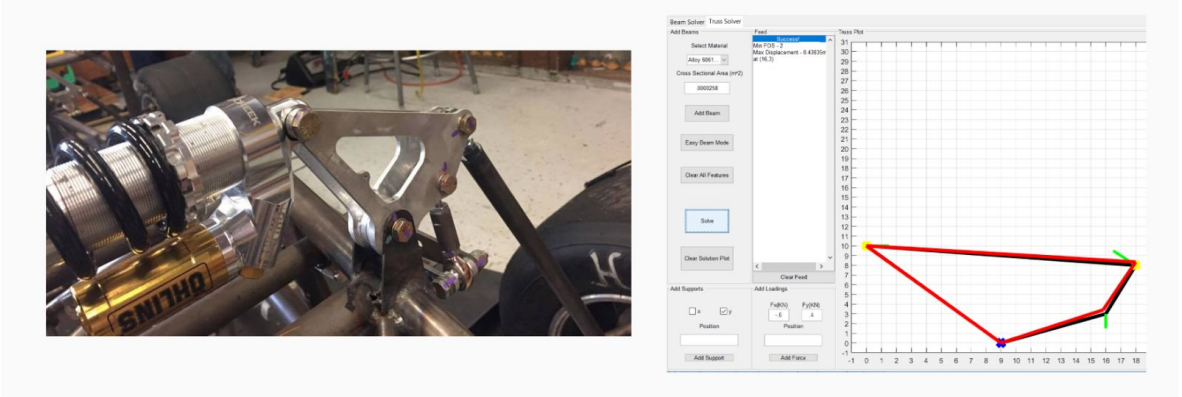
Beam Solver: Arbitrarily Difficult Loading Example



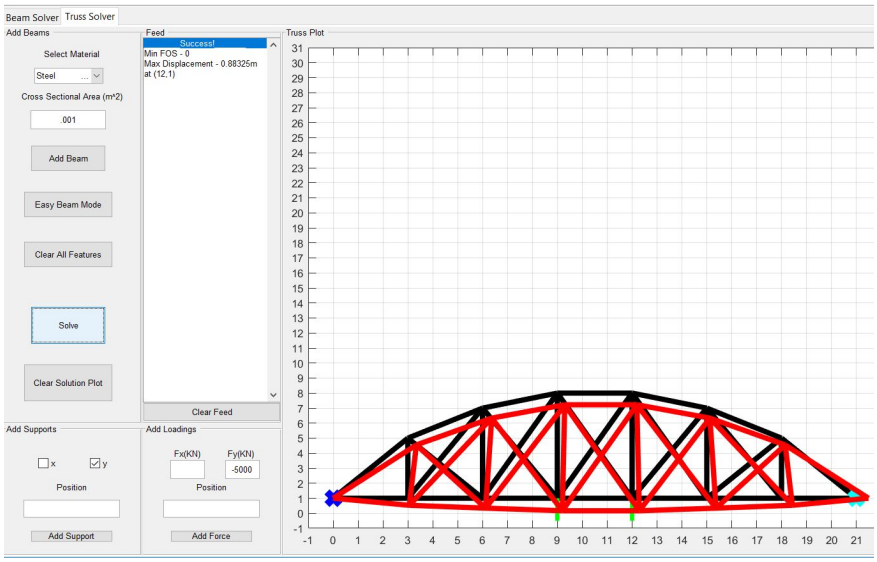
Truss Solver: FSAE Modeling Example 1



Truss Solver: FSAE Modeling Example 2



Truss Solver: Truss Example (exaggerated deformation)



Teammate Contributions

Albert Li was responsible for implementing the back-end of the beam solver, including the generation of the shear and moment diagrams. He generated analytical solutions to many common beam cases. In particular, he solved for almost all cases of beams with static indeterminacy of degree 2 or lower, and was able to successfully retrieve the shear and moment diagrams for those cases independent of difficult loading conditions. All the methods named “calculate…” were written by Albert, as well as all the methods in Beam and Load and all its subclasses. Albert also debugged the constructors for the Load and Support classes and subclasses in order to make them function correctly with the syntax of the back end.

Dominic Croce was responsible for implementing the GUI of the beam solver and mating it with the GUI for the truss solver, and formatted the menu and selection options for the best user experience. He was also responsible for retrieving, formatting, and loading in external material and beam-type data for use in the GUI and modifying and editing existing back-end code to provide further functionality for the front-end, such as removal of elements in the system or modification of their initially chosen values. Dominic also wrote the Support classes and debugged some of the issues involved with their declaration.

Jake Ramirez was responsible for implementing the truss solver, including the ability to solve truss systems in arbitrary shapes with a myriad of joint orientations. He also created the GUI for the truss solver, including the ability for a user to plot trusses from an easy-to-use graphical structure and visualize changes in the beam from various arbitrary loading conditions using a stiffness matrix that related forces and node displacements. Every class labelled “Truss” was written by him.